AN UNSTEADY FLOW-SOLVER WITH ALGEBRAIC GRID MOTION FOR AEROELASTIC SIMULATIONS

C. B. Allen
Department of Aerospace Engineering,
University of Bristol, Bristol, BS8 1TR, U.K.
Email: C.B.Allen@Bristol.ac.uk

Keywords: *Unsteady flow, Algebraic grid motion, Euler equations, Aeroelasticity*

Abstract

The moving and/or deforming surfaces resulting from aeroelastic simulations require deforming meshes, and it is common to use simple interpolation of surface displacements and velocities onto the initial undisturbed mesh. However, aeroelastic simulations can result in large displacements and deformations of solid surfaces, and simple interpolation of perturbations results in poor grid quality and possible grid crossover. A new grid motion technique is presented which is still simple in that it is driven solely by surface motion, but represents rotational effects near the solid surface, to maintain grid quality there. Significantly, the scheme is fully analytic, so is very cheap computationally and results in grid speeds also being available analytically. Results, in terms of unsteady grid motion and flow solution, show the scheme to be effective and efficient.

1 Introduction

Unsteady flow-solvers are now routinely used, and are being coupled with structural models to produce aeroelastic simulation codes, see for example [1, 2, 3]. Unsteady simulations will normally entail moving or deforming surfaces, and hence require moving or deforming meshes. If these deformations are small, or rotations are rigid, it is common to use simple interpolation methods to interpolate surface displacements and velocities onto the initial undisturbed mesh, or to rigidly rotate the entire mesh. For example oscillating aerofoils and wings have been simulated using both algebraic grid generation and motion [4, 5, 6, 7, 8] where a new grid is generated algebraically every time-step, and simple interpolation of surface displacements and velocities [9, 10, 11, 12, 13], where meshes are perturbed every time-step. The time-dependent motion of rotor blades in forward flight has been modelled using rigid rotations of the surface, and the meshes have been deformed using simple interpolation of surface displacements [9, 10, 11, 12, 14]. This has also been modelled using a multiblock approach with the inner (blade-fixed) blocks moving rigidly and the other blocks deformed with simple interpolation [15, 16].

Coupling a structural model with a CFD code in the time domain offers the opportunity to simulate aeroelastic response of wings and rotor blades [1, 2, 3, 17, 18], but the resulting surface motions may not be small. For example, the hinge motion plus elastic deformation of a rotor blade under loading in forward flight can result in large time-dependent displacements of the surface. Time-dependent grids cannot be computed by simple grid perturbation in this situation, as poor grid quality and grid crossover may result. Furthermore, once the surface deforms, rather than simply moving, rigid rotations of the mesh are also not possible. Hence, a more general

grid motion scheme has been developed and presented here, which maintains grid quality under large deformations and motions. To maintain grid quality under such conditions, particularly large rotations, some account needs to be taken of the surface normal direction changes to maintain the grid orthogonality at the surface. More advanced grid perturbation methods have been developed to include surface normal direction changes, such that changes to wing/blade geometries during design can be remeshed via perturbation instead of regenerating the entire mesh, see for example [19]. However, these methods require surface normal data, and it is expensive and awkward to recompute surface normals every time-step during an unsteady computation, and problems occur at non-smooth surface points, for example wing trailing edges. Hence, the scheme presented here is driven solely by the surface state, and requires no normal direction data. There has been previous work using only the surface state to generate instantaneous meshes, by adopting a spring analogy to model grid lines during unsteady computations [20, 21, 22, 23, 24]. In this approach the spring stiffness between each point is set to be proportional to the reciprocal of the line length, and when the surface moves the whole grid moves according to the spring stiffnesses. Hence, the unsteady mesh is driven solely by the surface state, and as the near surface mesh has the smallest cells, and hence line lengths, the inner mesh moves almost rigidly as required. This approach can also be applied to both structured [22, 23, 24] and unstructured [20, 21] meshes. However, the resulting set of equations determining the instantaneous position of every grid point is expensive to solve, and the grid positions are not available algebraically. This can lead to problems when evaluating the grid speeds required in the unsteady flow-solver.

A new interpolation is presented here, which is improved to maintain the grid quality, such that the near surface grid moves almost rigidly. This means the original grid quality is maintained. Significantly, the technique is analytic and so all grid positions and speeds at any time are available analytically. Details of the grid motion technique are first presented, followed by results of grid motion for a large elastic deformation of a high aspect ratio wing, and the effectiveness and efficiency of the scheme demonstrated. An efficient implicit unsteady inviscid upwind finite-volume flow-solver is also presented, and the solution resulting from the motion considered previously is presented.

2 Grid Motion Scheme

A wing of aspect ratio twelve, no twist or taper, and constant NACA0012 section is used as an example. The initial mesh is generated at zero incidence. The grid used for demonstration is a structured O-H mesh of dimensions 99(chord)$\times$65(span)$\times$32(vertical) points, with 48 sections on the wing. A transfinite interpolation method [25], is used to generate the mesh.

The coordinate system used is ξ in the chordwise direction, η in the spanwise direction, and ζ is the cordinate between the inner and outer boundary. The general transfinite interpolation method results in a recursive algorithm, see Eriksson [26], but this can be significantly reduced. The interpolation is performed here by

$$\underline{\mathbf{X}}(\xi,\eta,\zeta) = \psi_1^0(\zeta)\underline{\mathbf{X}}(\xi,\eta,0) + \psi_1^1(\zeta)\frac{\partial}{\partial\zeta}\underline{\mathbf{X}}(\xi,\eta,0)$$
$$+\psi_2^0(\zeta)\left[\psi_2^0(\zeta)\underline{\mathbf{X}}(\xi,\eta,1) + \psi_1^0(\zeta)\underline{\mathbf{X}_2}(\xi,\eta,1)\right] \quad (1)$$

where $\underline{\mathbf{X}}(\xi,\eta,0)$ is the inner boundary, $\underline{\mathbf{X}}(\xi,\eta,1)$ the outer boundary, and $\underline{\mathbf{X}_2}(\xi,\eta,1)$ a smoothed outer boundary function to control smoothness. It has been found that specifying one inner boundary derivative, and no outer boundary derivatives, gives sufficient control.

After generating the volume mesh an elliptic smoothing is applied [27]. The smoothing has been coded such that boundaries are also smoothed (see [12] for more details).

Figure 1 shows the wing surface mesh and the portside plane, the same planes plus wake and tip slits, and these planes plus the outer boundary. The outer boundary is set at 20 chords.

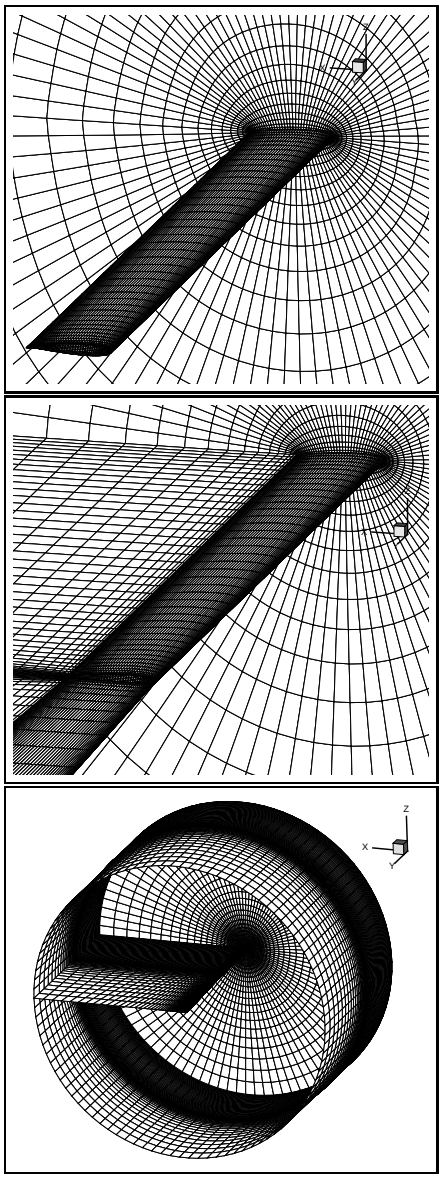

Fig. 1 *Blade Surface Mesh, Inner Boundary Planes, Inner and Outer Boundaries.*

2.1 Algebraic Motion Scheme

As in simple perturbation approaches, the instantaneous surface grid positions are used to compute the instantaneous displacement of each point from the initial mesh position. To account for the normal direction changes at the surface the displacement is split into two components: an unrotated displacement and a rotation. The rotation can be defined about any axis, but a spanwise axis is chosen here, as this is the axis about which the rotation is likely to be largest. This is performed by first computing the instantaneous rotation angle between the instantaneous surface and initial surface at each section, and this is labelled $\theta(\eta,t)$. It should be stated here that the deformation is not constrained to be about this axis, as will be demontrated later. To compute the unrotated displacement of the surface each section must first be rotated back through the angle $-\theta$. The rotation can be performed about any point, but the instantaneous centroid of the section, $\underline{\mathbf{X}}_c(\eta,t)$, is chosen to minimise the displacement effect of the rotation, and so the displacement of the surface is computed by

$$\Delta\underline{\mathbf{X}}(\xi,\eta,0,t) = \underline{\mathbf{X}}_c(\eta,t) + [\mathbf{R_Y}(-\theta)](\eta,t)\,\{\underline{\mathbf{X}}(\xi,\eta,0,t) - \underline{\mathbf{X}}_c(\eta,t)\} - \underline{\mathbf{X}}(\xi,\eta,0,0) \tag{2}$$

where $[\mathbf{R_Y}(\theta)]$ is the rotation matrix,

$$[\mathbf{R_Y}(\theta)] = \begin{bmatrix} \cos\theta & 0 & \sin\theta \\ 0 & 1 & 0 \\ -\sin\theta & 0 & \cos\theta \end{bmatrix} \tag{3}$$

The centroid is computed for each station as

$$\underline{\mathbf{X}}_c(\eta,t) = \frac{1}{imax}\sum_{i=1}^{imax}\underline{\mathbf{X}}(\xi,\eta,0,t) \tag{4}$$

The instantaneous grid position of every point can then be computed as a suitably weighted combination of this displacement and a rotation of $+\theta$ about the instantaneous centroid. This is achieved by

$$\underline{\mathbf{X}}(\xi,\eta,\zeta,t) = \underline{\mathbf{X}}(\xi,\eta,\zeta,0) + \psi_1(\zeta)\,\{\underline{\mathbf{X}}_c(\eta,t) + [\mathbf{R_Y}(-\theta)](\eta,t)\,\{\underline{\mathbf{X}}(\xi,\eta,0,t) - \underline{\mathbf{X}}_c(\eta,t)\} - \underline{\mathbf{X}}(\xi,\eta,0,0)\} + \psi_2(\zeta)\,\{[\mathbf{R_Y}(+\theta) - \mathbf{I}](\eta,t)\,\{\underline{\mathbf{X}}(\xi,\eta,\zeta,0) - \underline{\mathbf{X}}_c(\eta,0)\}\} \tag{5}$$

where the blending functions can be defined as required. For example, to move and rotate the grid effectively rigidly, ψ_1 and ψ_2 would both be unity. To reduce the displacement away from the surface these would normally satisfy

$$\psi_1(0) = 1.0 \qquad \psi_1(1) = 0.0 \tag{6}$$

$$\psi_2(0) = 1.0 \qquad \psi_2(1) = 0.0. \tag{7}$$

It is desirable that the near blade region moves/rotates rigidly, to maintain grid quality, while the outer boundary is fixed. Hence, the most effective blending functions have been found to be

$$\zeta = k/kmax \tag{8}$$

$$\overline{\zeta} = \left\{ \frac{e^{\zeta} - 1 - \zeta}{e-2} \right\}^2, \tag{9}$$

$$\psi_1 = 1 - \overline{\zeta}, \tag{10}$$

$$\psi_2 = 1 - \sin^4\left(\frac{\zeta\pi}{2}\right) \tag{11}$$

This technique has two major advantages. It is simple, and hence cheap, and is algebraic, so an algebraic equation can be derived for instantaneous speeds,

$$\frac{d}{dt}\underline{\mathbf{X}}(\xi,\eta,\zeta,t) = \psi_1(\zeta)\left\{ \frac{d}{dt}\underline{\mathbf{X}}_c(\eta,t) + \frac{d\theta}{dt}\frac{d}{d\theta}[\mathbf{R_Y}(-\theta)](\eta,t)\{\underline{\mathbf{X}}(\xi,\eta,0,t) - \underline{\mathbf{X}}_c(\eta,t)\} + [\mathbf{R_Y}(-\theta)](\eta,t)\left\{ \frac{d}{dt}\underline{\mathbf{X}}(\xi,\eta,0,t) - \frac{d}{dt}\underline{\mathbf{X}}_c(\eta,t) \right\} \right\} + \psi_2(\zeta)\left\{ \frac{d\theta}{dt}\frac{d}{d\theta}[\mathbf{R_Y}(+\theta) - \mathbf{I}](\eta,t)\{\underline{\mathbf{X}}(\xi,\eta,\zeta,0) - \underline{\mathbf{X}}_c(\eta,0)\} \right\} \tag{12}$$

The centroid velocity for each station is simply evaluated as

$$\frac{d}{dt}\underline{\mathbf{X}}_c(\eta,t) = \frac{1}{imax}\sum_{i=1}^{imax}\frac{d}{dt}\underline{\mathbf{X}}(\xi,\eta,0,t) \tag{13}$$

For a prescribed motion, every component in equation (12) is available, regardless of the surface motion. For a non-prescribed motion, the surface velocities would be available, and every component except the $\frac{d\theta}{dt}$ term known. However, from the surface velocities $\frac{d\theta}{dt}$ can be recovered, so an algebraic expression is always available for the grid speeds.

2.2 Test Case

An algebraic test case is used to demonstrate the scheme for large surface deformations. Oscillatory in-phase pitch and flap is used, with linear pitch angle variation from blade root to tip,

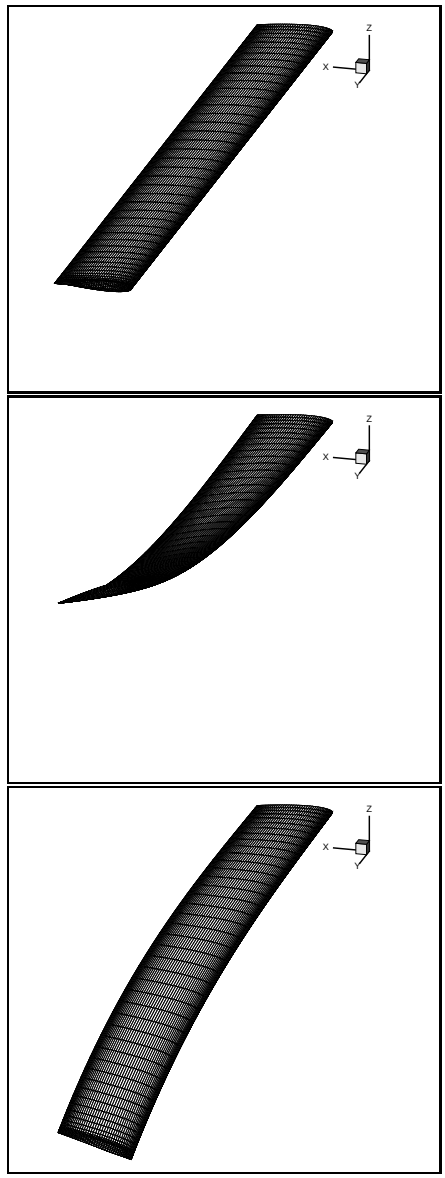

Fig. 2 *Blade Surface at* $\omega t = 0, \pi/2, 3\pi/2$.

$$\theta_Y(\eta,t) = \theta_{pitch}\left\{ \frac{\eta}{\eta_{Tip}} \right\}\sin(\omega t) \tag{14}$$

and a parabolic flap angle variation from root to tip,

$$\theta_X(\eta,t) = \theta_{flap}\left\{ \frac{\eta}{\eta_{Tip}} \right\}^2\sin(\omega t) \tag{15}$$

In the above, η is the spanwise coordinate and η_{Tip} is the coordinate of the tip.

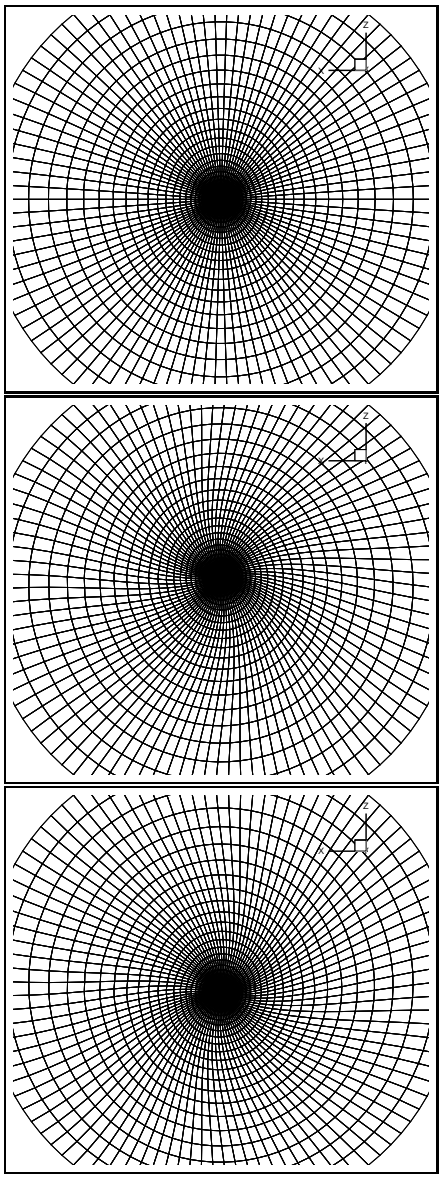

Fig. 3 *Grid plane at* $\omega t = 0, \pi/2, 3\pi/2$.

The time-dependent surface grid positions and velocities can be determined algebraically from equation (16).

$$\underline{\mathbf{X}}(\xi,\eta,0,t) = \underline{\mathbf{X}}_c(\eta-\Delta\eta,t)+$$

$$[\mathbf{R}_\mathbf{X}(\theta_X)](\eta,t)\,\{[\mathbf{R}_\mathbf{Y}(\theta_Y)](\eta,t)\,\{\underline{\mathbf{X}}(\xi,\eta,0,0)$$

$$-\underline{\mathbf{X}}_c(\eta,0)\}+\underline{\mathbf{X}}_c(\eta,0)-\underline{\mathbf{X}}_c(\eta-\Delta\eta,0)\} \quad (16)$$

where $[\mathbf{R}_\mathbf{X}(\theta)]$ is the rotation matrix about the x axis.

Results for pitch and flap angles of 20^o are shown. Figure 2 shows the extent of the blade surface deformation by showing its shape at $\omega t = 0, \pi/2$, and $3\pi/2$.

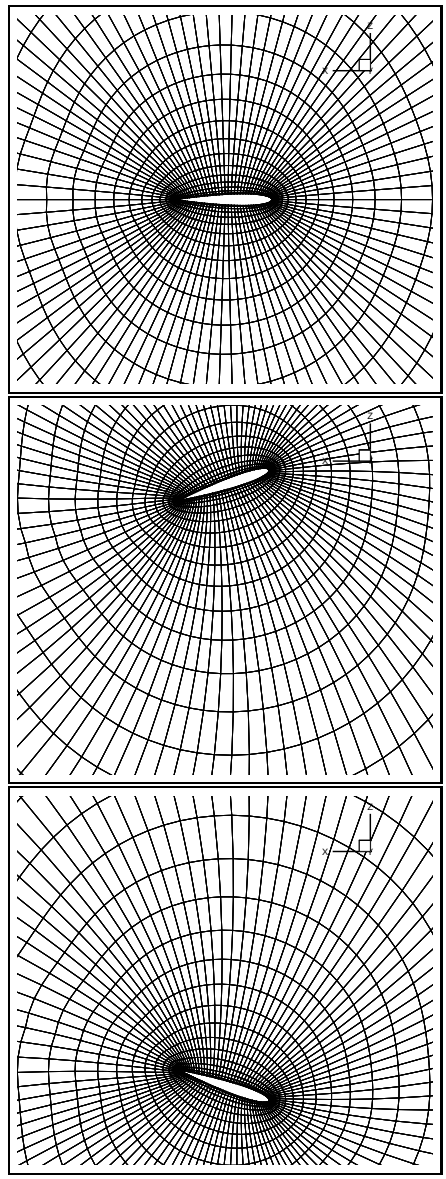

Fig. 4 *Grid plane at* $\omega t = 0, \pi/2, 3\pi/2$.

The effect of the deformation on the field grid is shown in figure 3, which shows a grid plane near the blade tip at $\omega t = 0, \pi/2$, and $3\pi/2$, and the near surface region of the same plane is shown in figure 4. These figures clearly demonstrate how the near surface mesh deforms almost rigidly, while the far field mesh is undeformed. The grid distributions obtained here are in fact very similar to those that would be obtained by solving the spring analogy equations, but are sig-

nificantly cheaper to compute, and are available algebraically.

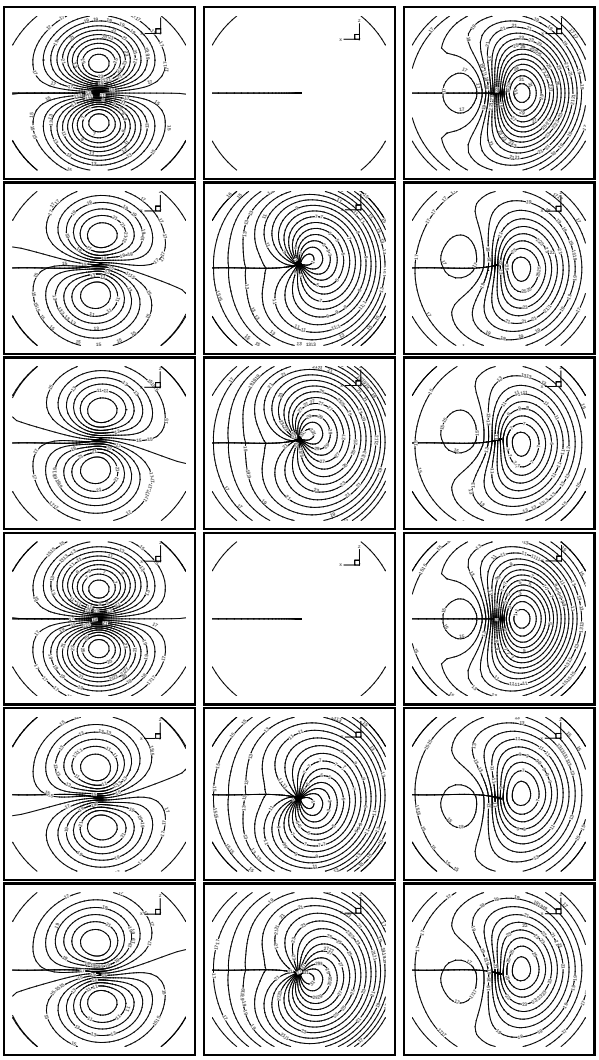

Fig. 5 *Grid Speed Contours,* $\omega t = 0, \pi/4, 3\pi/4, \pi, 5\pi/4, 7\pi/4$.

The grid positions are not frequency dependent, but the velocities are. An unsteady flow is computed in the next section, and a reduced frequency parameter,

$$k = \frac{\omega c}{2U_\infty} \tag{17}$$

where c is the wing chord and U_∞ the freestream velocity, of 0.1 is used. The Mach number used is 0.7 and so with a standard non-dimensionalisation, $\omega = 0.14$. Figure 5 shows dx/dt (left), dy/dt (middle), and dz/dt (right) contours at various points over one cycle for the same plane as shown in figures 3 and 4. All speeds are zero everywhere at $\omega t = \pi/2$ and $3\pi/2$, and so values are shown at $\omega t = 0, \pi/4, 3\pi/4, \pi, 5\pi/4, 7\pi/4$. All plots use 31 equally space contours. The grid plane shown is the last constant section plane on the blade surface, so has the largest grid speeds over the whole domain. This demonstrates the smooth and symmetric nature of the grid speeds.

3 Unsteady Euler Solver

The three-dimensional unsteady Euler equations in integral form are

$$\frac{d}{dt}\int_V \underline{\mathbf{U}} dV + \int_{\partial V} \underline{\mathbf{F}}.\underline{\mathbf{n}} dS = 0 \tag{18}$$

where

$$\underline{\mathbf{U}} = \begin{bmatrix} \rho \\ \rho u \\ \rho v \\ \rho w \\ E \end{bmatrix}, \quad \underline{\mathbf{F}} = \begin{bmatrix} \rho\left[\underline{\mathbf{q}} - \underline{\mathbf{X}}_\mathbf{t}\right] \\ \rho u\left[\underline{\mathbf{q}} - \underline{\mathbf{X}}_\mathbf{t}\right] + P\underline{\mathbf{i}} \\ \rho v\left[\underline{\mathbf{q}} - \underline{\mathbf{X}}_\mathbf{t}\right] + P\underline{\mathbf{j}} \\ \rho w\left[\underline{\mathbf{q}} - \underline{\mathbf{X}}_\mathbf{t}\right] + P\underline{\mathbf{k}} \\ E\left[\underline{\mathbf{q}} - \underline{\mathbf{X}}_\mathbf{t}\right] + P\underline{\mathbf{q}} \end{bmatrix}. \tag{19}$$

V is the computational volume, ∂V the volume surface, dS an element of the surface, $\underline{\mathbf{n}}$ the surface outward unit normal, and $\underline{\mathbf{q}}$ and $\underline{\mathbf{X}}_\mathbf{t}$ are the fluid and grid velocities respectively

$$\underline{\mathbf{q}} = \begin{bmatrix} u \\ v \\ w \end{bmatrix}, \quad \underline{\mathbf{X}}_\mathbf{t} = \begin{bmatrix} x_t \\ y_t \\ z_t \end{bmatrix}. \tag{20}$$

The equation set is closed by

$$P = (\gamma - 1)[E - \frac{\rho}{2}\underline{\mathbf{q}}^2]. \tag{21}$$

3.1 Upwind Difference Scheme

A finite-volume upwind scheme is used to solve the integral form of the Euler equations (equation 18). The flux-vector splitting of Van-Leer [28, 29] is used.

For each cell face a local orthogonal coordinate system (ξ,η,ζ) is adopted, where the principal coordinate direction ξ is normal to the cell face. The unit normal to each cell face is defined as the unit vector in the ξ direction, $\underline{\mathbf{n}}_\xi$. Unit vectors in two directions, lying in the cell face, $\underline{\mathbf{n}}_\eta$ and $\underline{\mathbf{n}}_\zeta$, are then defined to form an orthogonal axis system.

To compute the flux in the principal direction, the cartesian velocity components in the local cell face axis system are required,

$$\overline{u} = \underline{\mathbf{q}}.\underline{\mathbf{n}}_\xi, \tag{22}$$

$$\overline{v} = \underline{\mathbf{q}}.\underline{\mathbf{n}}_\eta, \tag{23}$$

$$\overline{w} = \underline{\mathbf{q}}.\underline{\mathbf{n}}_\zeta, \tag{24}$$

and the contravariant velocity normal to the face

$$\overline{U} = \left[\underline{\mathbf{q}} - \underline{\mathbf{X}_\mathbf{t}}\right].\underline{\mathbf{n}}_\xi. \tag{25}$$

The general flux function in the principal direction, $\underline{\overline{\mathbf{F}}}$, is then

$$\underline{\overline{\mathbf{F}}} = \begin{Bmatrix} \rho\overline{U} \\ \rho\overline{U}\overline{u}+P \\ \rho\overline{U}\overline{v} \\ \rho\overline{U}\overline{w} \\ E\overline{U}+P\overline{u} \end{Bmatrix} \tag{26}$$

and the total flux across the face is simply $\underline{\overline{\mathbf{F}}}A$, where A is the cell face area.

The general flux vector is split into a forward part, $\underline{\overline{\mathbf{F}}}^+$, associated with positive moving waves only, i.e. all eigenvalues of $\frac{\partial\underline{\overline{\mathbf{F}}}^+}{\partial\underline{\mathbf{U}}} \geq 0$, and a backward part, $\underline{\overline{\mathbf{F}}}^-$, associated with negative moving waves only, all eigenvalues of $\frac{\partial\underline{\overline{\mathbf{F}}}^-}{\partial\underline{\mathbf{U}}} \leq 0$. At each cell face a pair of states are thus defined and a single numerical flux derived from this pair. The split flux components are,

$$\underline{\overline{\mathbf{F}}}^\pm = \begin{Bmatrix} f^\pm_{mass} \\ f^\pm_{mass}.\left[\frac{(-\overline{U}\pm 2a)}{\gamma}+\overline{u}\right] \\ f^\pm_{mass}.\overline{v} \\ f^\pm_{mass}.\overline{w} \\ f^\pm_{energy} \end{Bmatrix} \tag{27}$$

where

$$f^\pm_{mass} = \pm\frac{\rho a}{4}(\overline{M}\pm 1)^2, \tag{28}$$

$$f^\pm_{energy} = f^\pm_{mass}\left\{\frac{[(\gamma-1)\overline{U}\pm 2a]^2}{2(\gamma^2-1)} - \frac{\overline{U}^2}{2}\right.$$

$$\left. + \frac{\underline{\mathbf{q}}^2}{2} + \underline{\mathbf{X}_\mathbf{t}}.\underline{\mathbf{n}}\frac{(-\overline{U}\pm 2a)}{\gamma}\right\}, \tag{29}$$

and the Mach number normal to the cell face is defined as

$$\overline{M} = \frac{\overline{U}}{a}, \tag{30}$$

$$a = \sqrt{\frac{\gamma P}{\rho}}. \tag{31}$$

The above splitting is only valid for $\left|\overline{M}\right| \leq 1$. Otherwise

$$\left.\begin{array}{l}\underline{\overline{\mathbf{F}}}^+ = \underline{\overline{\mathbf{F}}} \\ \underline{\overline{\mathbf{F}}}^- = 0\end{array}\right\} \quad \overline{M} > 1, \tag{32}$$

$$\left.\begin{array}{l}\underline{\overline{\mathbf{F}}}^+ = 0 \\ \underline{\overline{\mathbf{F}}}^- = \underline{\overline{\mathbf{F}}}\end{array}\right\} \quad \overline{M} < -1. \tag{33}$$

The values of the conserved variables used in the split fluxes must be consistent with the splitting, i.e. the positive vector must be evaluated using information from upstream (in the principal direction) of the cell face only, and the negative vector using information from downstream only. Hence the flux vector is split by

$$\underline{\overline{\mathbf{F}}} = \underline{\overline{\mathbf{F}}}^+(\underline{\mathbf{U}}^+) + \underline{\overline{\mathbf{F}}}^-(\underline{\mathbf{U}}^-), \tag{34}$$

with the upwind interpolations given by a third-order spatial interpolation [30]. High order schemes suffer from spurious oscillations in regions of high flow quantity gradients, and so a flux limiter is required, and the continuously differentiable one due to Anderson *et al* [30] was chosen.

Once $\underline{\overline{\mathbf{F}}}$ has been split into its components the resulting flux must be rotated back to the original coordinate system. This is achieved by

$$\underline{\mathbf{F}}.\underline{\mathbf{n}} = R^{-1}[\underline{\overline{\mathbf{F}}}^+(\underline{\mathbf{U}}^+) + \underline{\overline{\mathbf{F}}}^-(\underline{\mathbf{U}}^-)], \tag{35}$$

where R is the rotation matrix.

3.2 Implicit Time-Stepping Scheme

An efficient implicit scheme is adopted. The implicit form of the differential equation for each computational cell is considered,

$$\frac{\partial(V^{n+1}\underline{\mathbf{U}}^{n+1})}{\partial t} + \underline{\mathbf{R}}(\underline{\mathbf{U}}^{n+1}) = 0 \qquad (36)$$

where V is the time-dependent cell volume and $\underline{\mathbf{R}}$ is the upwinded flux integral. The implicit temporal derivative is then approximated by a second-order backward difference, following Jameson [31], and a new residual $\underline{\mathbf{R}}^*(\underline{\mathbf{U}})$ defined as

$$\underline{\mathbf{R}}^*(\underline{\mathbf{U}}) = \frac{3}{2\Delta t}\left[V^{n+1}\underline{\mathbf{U}}\right] - \frac{2}{\Delta t}\left[V^{n}\underline{\mathbf{U}}^{n}\right] + \frac{1}{2\Delta t}\left[V^{n-1}\underline{\mathbf{U}}^{n-1}\right] + \underline{\mathbf{R}}(\underline{\mathbf{U}}) \qquad (37)$$

and then a new differential equation can be written in terms of a fictitious time τ, (called pseudo-time). This equation is simply time-marched to convergence in the fictitious time τ, for each real time-step. For each real time step a multi-stage time-stepping scheme with local time-stepping is used in pseudo-time. The R.H.S. is manipulated such that it is implicit and gives a "residual" that tends to zero, see [8] for more details. For example integrating from pseudo time-level m to $m+1$, the scheme would be

$$\left(1+\alpha_j\frac{3\Delta\tau}{2\Delta t}\right)\underline{\mathbf{U}}^{m+\alpha_j} = \underline{\mathbf{U}}^{m} + \alpha_j\frac{3\Delta\tau}{2\Delta t}\underline{\mathbf{U}}^{m+\alpha_{j-1}} - \alpha_j\frac{\Delta\tau}{V^{n+1}}\left\{\frac{3}{2\Delta t}V^{n+1}\underline{\mathbf{U}}^{m+\alpha_{j-1}} - \frac{2}{\Delta t}V^{n}\underline{\mathbf{U}}^{n} + \frac{1}{2\Delta t}V^{n-1}\underline{\mathbf{U}}^{n-1} + \sum_{k=1}^{6} R_k^{-1}\left[\overline{\underline{\mathbf{F}}}^{+}(\underline{\mathbf{U}}^{+})_k^{m+\alpha_{j-1}} + \overline{\underline{\mathbf{F}}}^{-}(\underline{\mathbf{U}}^{-})_k^{m+\alpha_{j-1}}\right]A_k^{n+1}\right\} \qquad (38)$$

with $\alpha_{0,1,2,3} = 0, \frac{1}{4}, \frac{1}{2}, 1$. k represents the six cell faces, and $\Delta\tau$ is the pseudo time-step. The time step is now limited by accuracy rather than stability. This approach also means that the grid generation routine only needs to be called once every real time-step, to calculate the grid positions and speeds at the next time level. The time-dependent cell volumes are computed by satisfying a geometric conservation law [32].

4 Unsteady Solution

The solution to the previously shown prescribed motion was computed, using the following conditions,

$$M = 0.7, \quad k = 0.1, \qquad (39)$$

$$\theta_{flap} = \theta_{pitch} = 10^{o}. \qquad (40)$$

The steady solution at zero incidence was first computed then the unsteady motion started, and solution computed using the implicit scheme with 64 real time-steps per cycle.

Figure 6 shows upper surface Mach number contours over one cycle, at $\omega t = 0$, $\pi/4$, $\pi/2$, $3\pi/4$, π, $5\pi/4$, $3\pi/2$, and $7\pi/4$. (Sequence is left, right, etc. moving down the page) The growth and decay of a large transonic region is clear.

5 Conclusions

An algebraic grid motion technique suitable for large deformations has been presented, which is driven solely by the surface motion. The commonly used simple interpolation of surface displacements and velocities onto the initial mesh cannot be used for aeroelastic simulations resulting in large displacements/deformations of solid surfaces, as it results in poor grid quality and possible grid crossover. A new interpolation technique is presented which is still simple in that it is driven solely by surface motion, but represents rotational effects near the solid surface, to maintain grid quality there. Furthermore, the scheme is fully analytic, so is very cheap computationally and results in grid speeds also being available analytically. Since the grid positions and speeds are algebraic, they are independent of the time-step used in the unsteady simulation, which is not the case for a non-algebraic scheme, since numerical differences have to be used to compute the grid speeds. Results for a large surface deformation have shown the scheme to be effective at maintaining grid quality, and efficient.

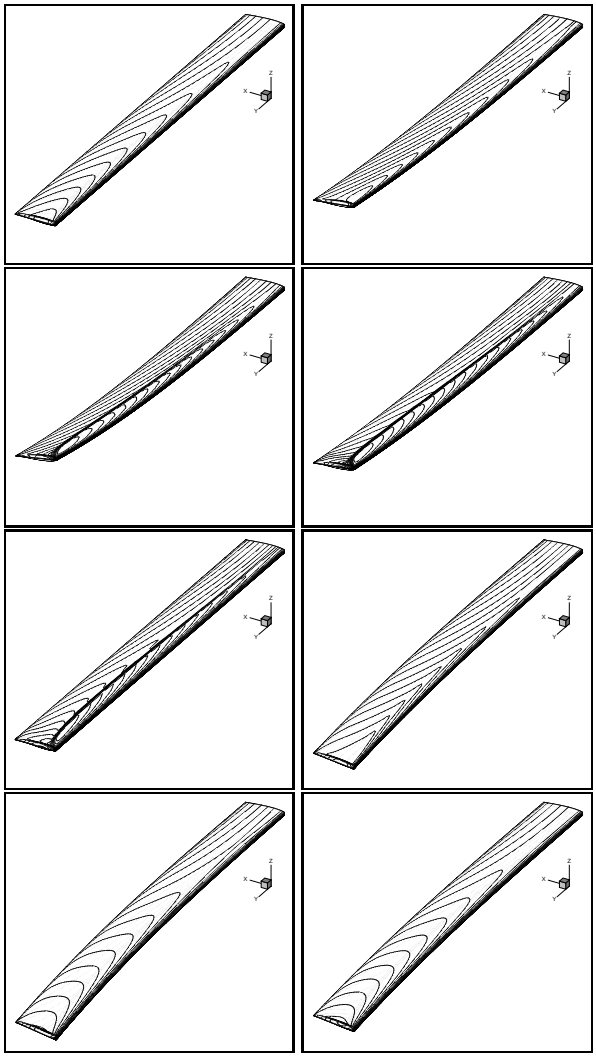

Fig. 6 *Upper surface Mach Contours at* $\omega t = 0, \pi/4, \pi/2, 3\pi/4, \pi, 5\pi/4, 3\pi/2, 7\pi/4$.

References

[1] Weeratunga, S.K. and Pramono, E., "*Direct Coupled Aeroelastic Analysis through Concurrent Implicit Time Integration on a Parallel Computer*", AIAA Paper 94-1550, 35th Structures, Structural Dynamics and Materials Conference, South Carolina, 1994.

[2] Farhat, C., Lesoinne, M., Chen, P.S. and Lanteri, S., "*Parallel Heterogeneous Algorithms for the Solution of Three-Dimensional Transient Coupled Aeroelastic Problems*", AIAA Paper 95-1290, AIAA/ASME/ASCE/AHS Structures, Structural Dynamics and Materials Conference, New York, 1995. Proceedings pp1146-1160.

[3] Piperno, S., Farhat, C. and Larroutorou, B., "*Partitioned Procedures for the Transient Solution of Coupled Aeroelastic Problems*", Computer Methods in Applied Mechanics and Engineering, vol. 124, pp79-112, June 1995.

[4] Gaitonde, A.L., " *A Dual-Time Method for the Solution of the unsteady Euler Equations* " The Aeronautical Journal of the Royal Aeronautical Society, Oct. 1994, pp 283-291

[5] Allen, C.B., "*Central-Difference and Upwind Biased Schemes for Steady and Unsteady Euler Aerofoil Flows*", Aeronautical Journal, Vol. 99, pp52-62, February 1995.

[6] Gaitonde, A.L. and Fiddes, S.P., "*A Comparison of a Cell-Centre Method and a Cell-Vertex Method for the Solution of the 2D Unsteady Euler Equations on a Moving Grid*", I.Mech.E Journal of Aerospace Engineering, Part G3, Vol 209, pp203-214, 1995.

[7] Allen, C. B., "*The Reduction of Numerical Entropy Generated by Unsteady Shockwaves*", Aeronautical Journal, Vol. 101, No. 1001, January 1997, pp9-16.

[8] Allen, C. B., "*Grid Adaptation for Unsteady Flow Computations*", I. Mech. E. Journal of Aerospace Engineering, Part G4, Vol. 211, 1997, pp237-250.

[9] Hounjet, M.H.L., Allen, C.B., Vigevano, L., Gasparini, L., Pagano, A., "*GEROS: A European Grid Generator for Rotorcraft Simulation Methods*", Presented at 6^{th} International Conference on Numerical Grid Generation in Computational Field Simulations, London, July 1998. Proceedings pp813-822.

[10] D'Alascio, A., Dubuc, L., Peshkin, D., Vigevano, L., Allen, C.B., Pagano, A., Salvatore, F., Boniface, J-C., Hounjet, M.H.L., Kroll, N., Scholl, E., Kokkalis, A., Righi, M., "*First Results of the EROS European Unsteady Euler Code on Overlapping Grids*", presented at ECCOMAS 98 Conference, Athens, September 1998.

[11] Renzoni, P., D'Alascio, A., Kroll, N., Peshkin, D., Hounjet, M.H.L., Boniface, J-C., Vigevano, L., Morino, L., Allen, C.B., Dubuc, L., Righi, M., Scholl, E., Kokkalis, A., "*EROS: A European Euler Code for Helicopter Rotor Simulations*", Journal of Progress in Aerospace Sci-

ences, 2000.

[12] Allen, C. B., "*CHIMERA Volume Grid Generation within the EROS Code*", I. Mech. E. Journal of Aerospace Engineering, Part G, 2000.

[13] Dubuc, L. et al, "*Solution of the Unsteady Euler Equations Using an Implicit Dual Time Method*", AIAA Journal, Vol. 36, 1998. pp1417-1424.

[14] Webster, R.S., Chen, J.P. and Whitfield, D.L., "*Numerical Solution of a Helicopter Rotor in Hover and Forward Flight*", AIAA Paper 95-0193, Reno, NV, January 1995.

[15] Pahlke, K. and Raddatz, J., "*3D Euler Methods for Multibladed Rotors in Hover and Forward Flight*", Paper 20, 19th European Rotorcraft Forum, Cernobbio (Como), Italy, September 1993.

[16] Boniface, J.-C., Mialon, B. and Sides, J., "*Numerical Simulation of Unsteady Euler Flow around Multibladed Rotor in Forward Flight using a Moving Grid Approach*", American Helicopter Society 51st Forum, Fort Worth, TX, May 1995.

[17] Wagner, S., Altmikus, A.R.M., Hablowetz, T. and Well, K.H., "*On the Accuracy of Modular Aeroelastic Methods Applied to Fixed and Rotary Wings*", AIAA Paper 2000-4224, Applied Aerodynamics Conference, Denver, CO., August 2000. Proceedings pp436-449.

[18] Buchtala, B., Wehr, D. and Wagner, S., "*Coupling of Aerodynamic and Dynamic Methods for the Calculation of Helicopter Rotors in Forward Flight*", 23rd European Rotorcraft Forum, pp5.1-5.12, Dresden, September 1997.

[19] Allen, C.B., "*Efficient Use of an Upwind Euler Code for Multi-Bladed Rotor Design*", AIAA Paper 99-3227, presented at 17^{th} AIAA Applied Aerodynamics Conference, Norfolk, Virginia, June 1999. Proceedings pp830-840.

[20] Batina, J.T., "*Unsteady Euler Airfoil Solutions Using Unstructured Dynamic Meshes*", AIAA Paper 89-0115, Reno, NV, January 1989.

[21] Batina, J.T., "*Unsteady Euler Algorithm with Unstructured Dynamic Mesh for Complex-Aircraft Aerodynamic Analysis*", AIAA Journal, Vol. 29, No. 3, March 1991. pp327-333.

[22] Nakahashi, K. and Deiwert, G.S., "*Self-Adaptive-Grid Method with Application to Airfoil Flow*", AIAA Journal, Vol. 25, 1987. pp513-520.

[23] Prananta, B.B., Hounjet, M.H.L. and Zwaan, R.J., "*A Thin-Layer Navier-Stokes Solver and it's Applications for Aeroelastic Analysis of an Aerofoil in Transonic Flows*", Proceedings of 1995 International Forum on Aeroelasticity and Structural Dynamics, London June 1995. pp15.1-15.15.

[24] Prananta, B.B. and Hounjet, M.H.L., "*Aeroelastic Simulation with Advanced CFD Methods in 2D and 3D Transonic Flow*", Proceedings RAeS Conference on Unsteady Aerodynamics, London June 1996.

[25] Gordon, W. J. and Hall, C. A., "*Construction of Curvilinear Coordinate Systems and Applications of Mesh Generation*", International Journal of Numerical Methods in Engineering, Vol. 7, 1973, pp. 461-477.

[26] Eriksson, L. E., "*Generation of Boundary-Conforming Grids Around Wing-Body Configurations Using Transfinite Interpolation*", AIAA Journal, Vol. 20, No. 10, 1982, pp. 1313-1320.

[27] Thompson, J. F., "*A General Three Dimensional Elliptic Grid Generation System on a Composite Block-Structure*", Computer Methods in Applied Mechanics and Engineering, Vol. 64, 1987, pp. 377-411.

[28] Van-Leer, B., "*Flux-Vector Splitting for the Euler Equations*", Lecture Notes in Physics, Vol. 170, 1982, pp. 507-512.

[29] Parpia, I. H., "*Van-Leer Flux-Vector Splitting in Moving Coordinates*", AIAA Journal, Vol. 26, January 1988, pp. 113-115.

[30] Anderson, W. K. Thomas, J. L. and Van-Leer, B., "*Comparison of Finite Volume Flux Vector Splittings for the Euler Equations*", AIAA Journal, Vol. 24, September 1986, pp. 1453-1460.

[31] Jameson, A., "Time Dependent Calculations Using Multigrid, with Applications to Unsteady Flows Past Airfoils and Wings", AIAA Paper 91-1596.

[32] Thomas, P. D. and Lombard, C. K., "Geometric Conservation Law and its Application to Flow Computations on Moving Grids", AIAA J, Vol. 17, October 1979, pp. 1030-1037.